

```

1 %% Machine Learning
2 % Lab 9: Support Vector Machine (SVM)
3 % — Classification —
4
5 %% Initialization
6 clear ; close all; clc
7
8 %% ===== Part 1: Loading and Visualizing Data =====
9 % We start the exercise by first loading and visualizing the dataset.
10 % The following code will load the dataset into your environment and plot
11 % the data.
12 %
13
14 fprintf('Loading and Visualizing Data ...\n')
15
16 % Load from ex6data1:
17 % You will have X, y in your environment
18 load('ex6data1.mat');
19
20 % Plot training data
21 plotData(X, y);
22
23 fprintf('Program paused. Press enter to continue.\n');
24 pause;
25
26 %% ===== Part 2: Training Linear SVM =====
27 % The following code will train a linear SVM on the dataset and plot the
28 % decision boundary learned.
29 %
30
31 % Load from ex6data1:
32 % You will have X, y in your environment
33 load('ex6data1.mat');
34
35 fprintf('\nTraining Linear SVM ...\n')
36
37 % You should try to change the C value below and see how the decision
38 % boundary varies (e.g., try C = 1000)
39 C = 0.5;
40 model = svmTrain(X, y, C, @linearKernel, 1e-3, 20);
41 visualizeBoundaryLinear(X, y, model);
42
43 fprintf('Program paused. Press enter to continue.\n');
44 pause;
45 %% ===== Part 3: Implementing Gaussian Kernel =====
46 % You will now implement the Gaussian kernel to use
47 % with the SVM. You should complete the code in gaussianKernel.m
48 %
49 fprintf('\nEvaluating the Gaussian Kernel ...\n')
50
51 x1 = [1 2 1]; x2 = [0 4 -1]; sigma = 2;
52 sim = gaussianKernel(x1, x2, sigma);
53
54 fprintf(['Gaussian Kernel between x1 = [1; 2; 1], x2 = [0; 4; -1], sigma = %f : '
55         ...
56         '\n\t%f\n(for sigma = 2, this value should be about 0.324652)\n'], sigma,

```

```

sim);
56
57 fprintf('Program paused. Press enter to continue.\n');
58 pause;
59
60 %% ===== Part 4: Visualizing Dataset 2 =====
61 % The following code will load the next dataset into your environment and
62 % plot the data.
63 %
64
65 fprintf('Loading and Visualizing Data ...\n')
66
67 % Load from ex6data2:
68 % You will have X, y in your environment
69 load('ex6data2.mat');
70
71 % Plot training data
72 plotData(X, y);
73
74 fprintf('Program paused. Press enter to continue.\n');
75 pause;
76
77 %% ===== Part 5: Training SVM with RBF Kernel (Dataset 2) =====
78 % After you have implemented the kernel, we can now use it to train the
79 % SVM classifier.
80 %
81 fprintf('\nTraining SVM with RBF Kernel (this may take 1 to 2 minutes) ...\n');
82
83 % Load from ex6data2:
84 % You will have X, y in your environment
85 load('ex6data2.mat');
86
87 % SVM Parameters
88 C = 1; sigma = 0.1;
89
90 % We set the tolerance and max_passes lower here so that the code will run
91 % faster. However, in practice, you will want to run the training to
92 % convergence.
93 model= svmTrain(X, y, C, @(x1, x2) gaussianKernel(x1, x2, sigma));
94 visualizeBoundary(X, y, model);
95
96 fprintf('Program paused. Press enter to continue.\n');
97 pause;
98
99 %% ===== Part 6: Visualizing Dataset 3 =====
100 % The following code will load the next dataset into your environment and
101 % plot the data.
102 %
103
104 fprintf('Loading and Visualizing Data ...\n')
105
106 % Load from ex6data3:
107 % You will have X, y in your environment
108 load('ex6data3.mat');
109
110 % Plot training data
111 plotData(X, y);

```

```

112
113 fprintf('Program paused. Press enter to continue.\n');
114 pause;
115
116 %% ===== Part 7: Training SVM with RBF Kernel (Dataset 3) =====
117
118 % This is a different dataset that you can use to experiment with. Try
119 % different values of C and sigma here.
120 %
121
122 % Load from ex6data3:
123 % You will have X, y in your environment
124 load('ex6data3.mat');
125
126 % Try different SVM Parameters here
127 [C, sigma] = dataset3Params(X, y, Xval, yval);
128
129 % Train the SVM
130 model= svmTrain(X, y, C, @(x1, x2) gaussianKernel(x1, x2, sigma));
131 visualizeBoundary(X, y, model);
132
133 fprintf('Program paused. Press enter to continue.\n');
134 pause;

```

linearKernel.m

```
1 function sim = linearKernel(x1, x2)
2 %LINEARKERNEL returns a linear kernel between x1 and x2
3 %   sim = linearKernel(x1, x2) returns a linear kernel between x1 and x2
4 %   and returns the value in sim
5
6 % Ensure that x1 and x2 are column vectors
7 x1 = x1(:); x2 = x2(:);
8
9 % Compute the kernel
10 sim = x1' * x2; % dot product
11
12 end
```

gaussianKernel.m

```
1 function sim = gaussianKernel(x1, x2, sigma)
2 %RBFKERNEL returns a radial basis function kernel between x1 and x2
3 %   sim = gaussianKernel(x1, x2) returns a gaussian kernel between x1 and x2
4 %   and returns the value in sim
5
6 % Ensure that x1 and x2 are column vectors
7 x1 = x1(:); x2 = x2(:);
8
9 % You need to return the following variables correctly.
10 sim = 0;
11
12 sim = exp(-(sum((x1-x2).^2)/(2*sigma^2)));
13
14 end
```